

Article

A Novel Deep Learning Method for Detecting Strawberry Fruit

Shuo Shen ¹, Famin Duan ^{2,*}, Zhiwei Tian ^{2,*}  and Chunxiao Han ³
¹ Faculty of Electronic Information and Electrical Engineering, Dalian University of Technology, Dalian 116024, China

² Institute of Urban Agriculture, Chinese Academy of Agricultural Sciences, Chengdu 610213, China

³ SDIC Xinjiang Luobupo Potash Co., Ltd., Hami 839000, China

* Correspondence: duanfamin@caas.cn (F.D.); tianzhiwei@caas.cn (Z.T.)

Abstract: The recognition and localization of strawberries are crucial for automated harvesting and yield prediction. This article proposes a novel RTF-YOLO (RepVgg-Triplet-FocalLoss-YOLO) network model for real-time strawberry detection. First, an efficient convolution module based on structural reparameterization is proposed. This module was integrated into the backbone and neck networks to improve the detection speed. Then, the triplet attention mechanism was embedded into the last two detection heads to enhance the network's feature extraction for strawberries and improve the detection accuracy. Lastly, the focal loss function was utilized to enhance the model's recognition capability for challenging strawberry targets, which thereby improves the model's recall rate. The experimental results demonstrated that the RTF-YOLO model achieved a detection speed of 145 FPS (frames per second), a precision of 91.92%, a recall rate of 81.43%, and an *mAP* (mean average precision) of 90.24% on the test dataset. Relative to the baseline of YOLOv5s, it showed improvements of 19%, 2.3%, 4.2%, and 3.6%, respectively. The RTF-YOLO model performed better than other mainstream models and addressed the problems of false positives and false negatives in strawberry detection caused by variations in illumination and occlusion. Furthermore, it significantly enhanced the speed of detection. The proposed model can offer technical assistance for strawberry yield estimation and automated harvesting.

Keywords: object detection; YOLOv5; structural reparameterization; feature enhancement



Citation: Shen, S.; Duan, F.; Tian, Z.; Han, C. A Novel Deep Learning Method for Detecting Strawberry Fruit. *Appl. Sci.* **2024**, *14*, 4213. <https://doi.org/10.3390/app14104213>

Academic Editors: Salik Khanal and Vitor Filipe

Received: 27 March 2024

Revised: 11 May 2024

Accepted: 13 May 2024

Published: 16 May 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Strawberry is among the most popular small berries worldwide due to its high economic and nutritional values [1]. However, strawberry fruit matures quickly and is susceptible to damage, which can easily result in decay and economic loss if not harvested promptly. Currently, strawberry harvesting relies on traditional manual approaches, which causes problems of high labor intensity and costs [2]. Automated strawberry harvesting can effectively improve productivity and reduce labor costs, which presents a promising solution to these challenges.

Strawberry detection is a crucial technology for achieving automated harvesting. Traditionally, the primary detection methods involve the integration of machine learning algorithms and computer vision techniques. Image-processing techniques, such as gray morphology, threshold segmentation, logical operations, the OTSU (Nobuyuki Otsu) method, and mean shift segmentation, are widely employed to extract the visual characteristics of fruits from various color spaces [3,4]. Subsequently, the extracted visual characteristics are combined with prediction algorithms, such as Kalman filtering, and machine learning algorithms, such as K-nearest neighbor and SVMs (support vector machines), to facilitate target recognition and localization [5–7]. While the mentioned methods can effectively identify fruit targets in certain environments, their performance and robustness may be affected by fluctuating lighting and fruit occlusion [8].

Currently, deep neural network methods have gained popularity for strawberry detection due to their robust feature extraction capabilities [9]. Applying deep neural networks with high accuracy and speed for strawberry harvesting machinery has drawn much research attention. Zhang et al. [10] proposed an improved lightweight network RTSD-Net based on YOLOv4-Tiny for strawberry target detection under field conditions. Similarly, Yu et al. [11] presented a novel strawberry detection method implemented on a harvesting robot to localize picking points based on the lightweight network Mobilenet-V1. Mejia et al. [12] proposed a strawberry localization method and an autonomous rover system, which utilizes image-processing techniques and the Mobilenet-V2 network to estimate strawberry ripeness in field conditions. Originally designed for embedded devices, these models prioritized lightweight design and detection speed. However, this emphasis compromised their accuracy in complex environments, which was only approximately 80%.

Many researchers employed two-stage detection algorithms, which are typically represented by the R-CNN (region-based convolutional neural network) series, to enhance strawberry detection accuracy. Yu et al. [13] proposed a novel algorithm to achieve the visual localization of strawberry picking points based on the Mask R-CNN algorithm, which demonstrated robustness for overlapping and occluded fruits under various lighting conditions. Tang et al. [14] proposed a strawberry detection method that combines Mask R-CNN, region segmentation techniques, and SVM classifiers, which can accurately detect strawberries at six different ripeness levels. Perez-Borrero et al. [15] proposed a specialized Mask R-CNN model for strawberry instance segmentation, which promised both speed and precision outcomes. Chen et al. [16] utilized a UAV to capture near-surface photographs of strawberry fields and employed the Faster R-CNN algorithm to detect and count flowers, mature strawberries, and immature strawberries, which achieved an average counting accuracy of 84.1%. Also, Zheng et al. [17] proposed a detection model based on the Faster R-CNN algorithm for counting strawberry fruit and flowers and effectively addressed the occlusion problem. Zhou et al. [2] proposed a Faster-RCNN-based strawberry-monitoring system that uses image acquisition and transfer learning to detect target fruits and measure strawberry ripeness and achieved an average accuracy exceeding 86%. Li et al. [18] introduced an intelligent system that uses Faster R-CNN to detect strawberries in field scenarios, which achieved high accuracy for the automatic monitoring and harvesting of strawberries. While these R-CNN-based models demonstrated high detection accuracies, their two-stage processing typically led to slower operation speeds, averaging around 15 FPS.

To balance detection accuracy and speed, single-stage detection algorithms have been widely employed in strawberry detection. The YOLO series is a typical one-stage detection algorithm with a speed advantage achieved by directly predicting classifications and bounding boxes. Wang et al. [19] proposed a DSE (detailed semantics enhancement) model based on YOLOv3 for multi-stage strawberry detection, which excelled in identifying different strawberry stages, with an average precision of 86.58%. However, the model's large size, with 300 million parameters, resulted in a slower detection speed of just 18 FPS and hindered deployment on embedded devices. Du et al. [8] proposed a DSW-YOLO(DCNv3-SA-WIoU-YOLO) network model for detecting strawberries at different occlusion levels, which achieved an average precision of 86.7%. Zhou et al. [20] introduced a YOLOv3-based method for classifying strawberry maturity in UAV (unmanned aerial vehicle) images, which achieved an average accuracy of 88%. The aforementioned YOLO-based models have effectively improved the detection speed, but there is still room for improvement in the detection accuracy. Chai et al. [21] proposed a novel strawberry detection algorithm based on a unique combination of YOLOv7 and augmented reality technology, which achieved a high F1 score of 92%. However, the method's accuracy in natural environments cannot be guaranteed, as the detection environment was limited to a greenhouse. Li et al. [22] proposed a multi-stage strawberry detection algorithm by integrating the ASFF (adaptive spatial feature fusion) module into YOLOv5, which improved the detection performance in natural environments, with an average precision exceeding 90%. However, this integration also led to a significant decrease in detection speed.

In summary, most research focused on improving specific performance metrics without achieving a well-balanced trade-off between model accuracy, speed, and size. To address this problem, an RTF-YOLO network model based on YOLOv5 is proposed in this paper to detect strawberry fruits. The main contributions of this work are summarized as follows:

- (1) A novel neural network model was proposed for strawberry detection under varying illumination and occlusion scenarios, which demonstrated improved speed and accuracy compared with other mainstream networks.
- (2) An efficient convolution module based on structural reparameterization was proposed and fused into the backbone and neck networks, improving the model's detection speed from 122 to 145 FPS.
- (3) The triplet attention mechanism and focal loss function were introduced to improve the detection precision, which led to a 3.6% increase in the $mAP_{0.5}$, reaching 90.24%.

The rest of this article is organized as follows. Section 2 introduces the specific algorithm, while Section 3 focuses on the results of the conducted experiments to evaluate and compare the proposed model with state-of-the-art methods. Section 4 summarizes the main conclusions of this work.

2. Materials and Methods

2.1. Image Acquisition And Augmentation

In this study, the growth stages of strawberries were observed, and images were captured using a camera. We collected a total of 2040 images, which were split into training, validation, and testing datasets with proportions of 70%, 20%, and 10%, respectively. These images were labeled as ripen and unripen, with a total count of 8712 strawberry targets, as shown in Table 1.

In natural environments, illumination variations and occlusion instances are the main factors that affect the detection performance. The dataset included samples under such scenarios to enhance the model's generalization performance, as shown in Figure 1.

The data were augmented with the following methods: horizontal flipping, rotating by 90 degrees, rotating by 180 degrees, rotating by 270 degrees, and randomly changing the saturation and brightness [23], as shown in Figure 2. The number of images increased to 9180 after the data augmentation.

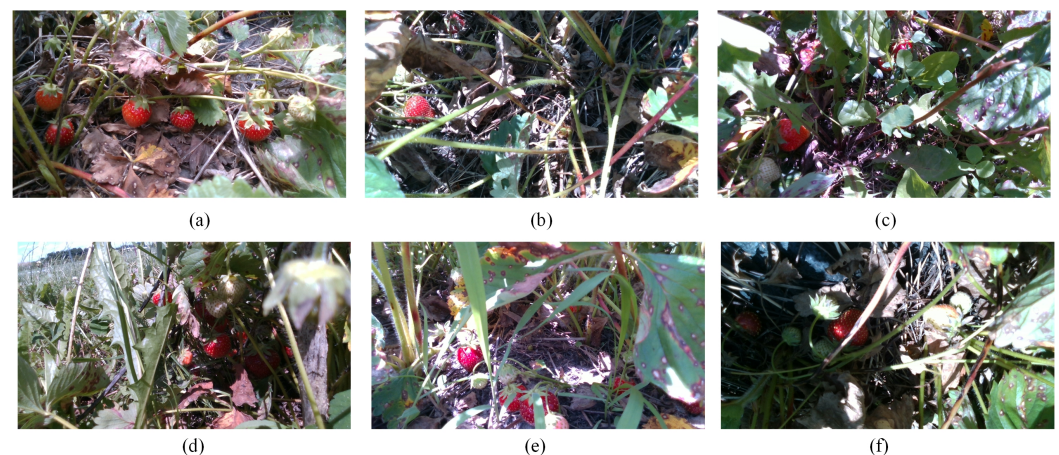


Figure 1. Images under different lighting and occlusion scenarios: (a) normal scenario, (b,c) occlusion scenarios, (d) low-light scenario, (e) high-light scenario, and (f) mixed low-light and high-light scenario.

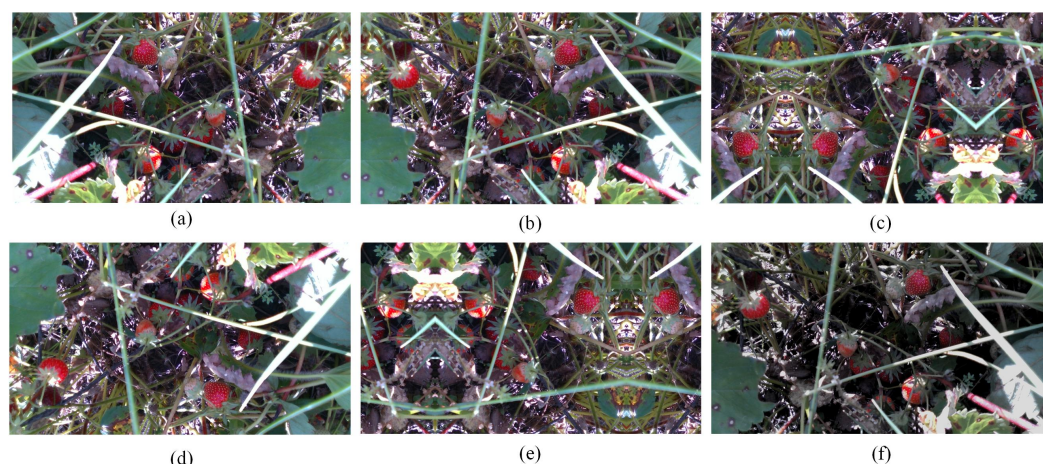


Figure 2. Image augmentation methods: (a) original image, (b) horizontal mirror, (c) 90° clockwise rotation, (d) 180° clockwise rotation, (e) 270° clockwise rotation, and (f) random transformation of saturation and brightness.

Table 1. Details of the strawberry dataset used for training, validating, and testing the proposed model.

Dataset	No. Images	Label		
		Ripen	Unripen	Total
Training	1428	3240	2952	6192
Validation	408	962	738	1700
Test	204	370	450	820

2.2. RFT-YOLO for Detecting Strawberry

This paper proposes a novel RTF-YOLO model derived from YOLOv5. The framework of RTF-YOLO is shown in Figure 3, and the specific module structures of YOLOv5 are shown in Figure 4.

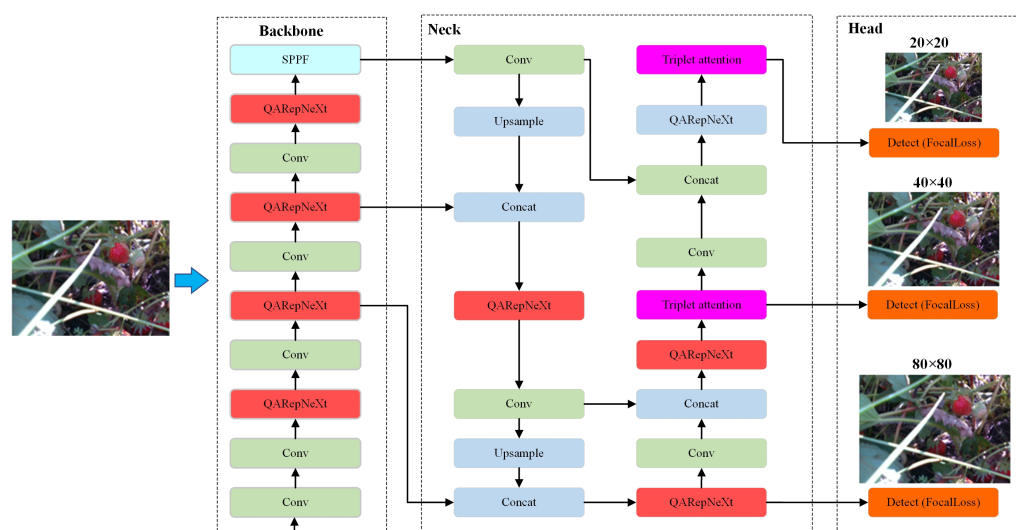


Figure 3. The overall framework of the RTF-YOLO model. The backbone network consists of QARepNeXt, Conv, and SPPF modules. The neck network consists of FPN and PAN structures and the triplet attention mechanism. The head network includes three detection heads with different resolutions and utilizes the focal loss function for target classification.

YOLOv5 is a convolutional neural network mainly comprising three parts: a backbone network, a neck network, and a detection head. The backbone network consists of the Conv,

C3 (cross-stage partial), and SPPF (spatial pyramid pooling—fast) modules. The Conv and C3 modules are the primary feature extractors, while the SPPF is employed to extract multi-scale features to improve the detection accuracy for objects of varying sizes.

YOLOv5s's neck part was constructed by PANet, which employs a hierarchical FPN (feature pyramid network) structure [24] and PAN (path aggregation network) structure [25] to transmit distinctive semantic attributes and positional characteristics from top to bottom and vice versa. PANet enhances the receptive field and provides a richer input representation. YOLOv5s's detection head comprises three convolutional layers to detect objects at distinct scales. This multi-scale design, coupled with varied aspect ratio anchor boxes, enhances the detection performance across diverse object sizes and shapes. YOLOv5 utilizes a compound loss function, which comprises localization loss, confidence loss, and classification loss. The CE (cross-entropy) loss and GIOU (generalized intersection over union) method were selected for the classification loss and localization loss, respectively.

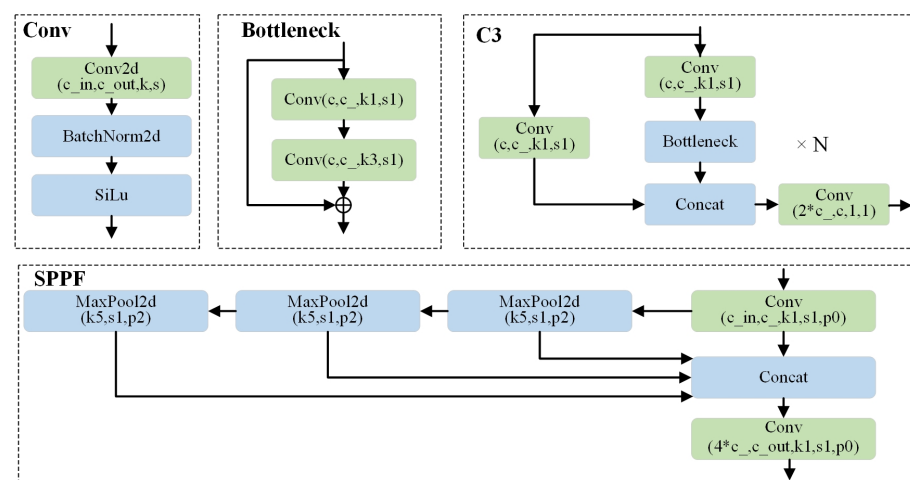


Figure 4. The C3 and SPPF modules of YOLOv5. The C3 combines N bottleneck blocks with three Conv modules, which optimizes the feature fusion for efficient feature extraction. The SPPF module generates multi-scale feature maps by applying max pooling at different scales, which enriches the semantic information of the feature representation.

The RTF-YOLO model essentially follows YOLOv5s's architecture due to its superiority and enhances the backbone network, neck network, and loss function. First, the C3 module was replaced by the proposed QARepNeXt module to reduce the computational complexity based on structure reparameterization technology. Then, the triplet attention mechanism was embedded in the neck network to enhance feature fusion. Lastly, the focal loss function was introduced to replace the CE loss function, which can balance the positive and negative examples and strengthen the learning ability of challenging examples.

2.2.1. Backbone

The backbone network was reconstructed with structure reparameterization technology [26] and the QARepVgg block [27].

Many mainstream studies focused on well-designed architectures like ResNet, MobileNet, and Inception. These networks use multi-branch parallelism to increase the model's representational capacity and improve the accuracy. However, complex multi-branch designs lead to increased memory access and usage, which, in turn, reduces the model's inference speed [26,28].

Ding et al. [26] proposed the RepVgg block based on the structure reparameterization technology. The RepVgg block has different structures during training and inferencing. In training, the RepVgg Block has three parallel branches, namely, the 3×3 convolution, 1×1 convolution, and Identity branches, with each followed by a BN (batch normaliza-

tion) layer. During inference, the three parallel branches are converted into a single-path structure, as shown in Figure 5a, specifically:

- (1) In the 1×1 convolution branch, the 1×1 convolution is converted into an equivalent 3×3 convolution by padding.
- (2) In the identity branch, the identity is converted into an equivalent 1×1 convolution by using the identity matrix as the kernel. Then, the 1×1 convolution is converted into a 3×3 convolution by padding.
- (3) All three branches are converted into the structure of a 3×3 convolution followed by a BN layer. Then, the 3×3 convolution and BN layer are fused into a single 3×3 convolution on each branch.
- (4) The 3×3 convolutions of the three branches are merged into one branch through addition.

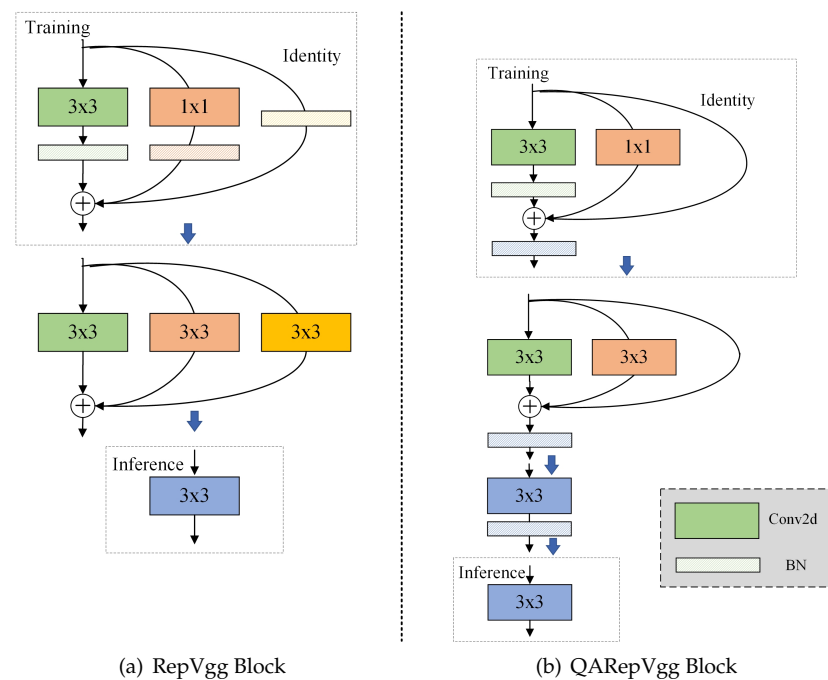


Figure 5. Reparameterization of a QARepVgg block compared with RepVgg. The main improvements were as follows: (1) the BN layers followed by both the 1×1 branch and identity branches were removed, and (2) a new BN layer was appended to the convolutional layer obtained from the fusion of multiple branches.

The RepVgg block achieved an optimal balance between performance and inference speed. However, in quantization scenarios, models with the RepVgg Block experienced a significant decline in performance, including an over 20% top one accuracy drop on ImageNet with INT8 (8-bit integer) inference.

Chu et al. [27] conducted an in-depth analysis of performance degradation during the standard quantization. The authors introduced a simple yet effective improved structure, QARepVgg block, which benefited from the advantages of reparameterization and possessed quantization-friendly attributes, as illustrated in Figure 5b. Consequently, networks based on the QARepVgg block significantly minimize the performance discrepancy between INT8 and FP32 (32-bit floating point), which enhances the model's suitability for deployment on edge devices.

We utilized the QARepVgg block to create the QARepNeXt module. In detail, the width (the size of the feature map) of the QARepNeXt module was kept consistent with that of the C3 module, and the stacking layers of the QARepVgg block were aligned with the BottleNeck in the C3 module. Similar to RepVgg, QARepNeXt operates as a multi-branch parallel structure during training and converts to a single-path structure during inference,

as depicted in Figure 6. The multi-branch structure promotes diverse feature learning and robustness during training but increases the computational complexity. Converting to a single-path structure can enhance the inference speed while preserving the benefits of the features learned during training.

The QARepNeXt module was implemented to replace the C3 module in the YOLOv5 model, which resulted in an enhanced backbone and neck network. This substitution led to a 26% elevation in inference speed while maintaining the model's original performance quality.

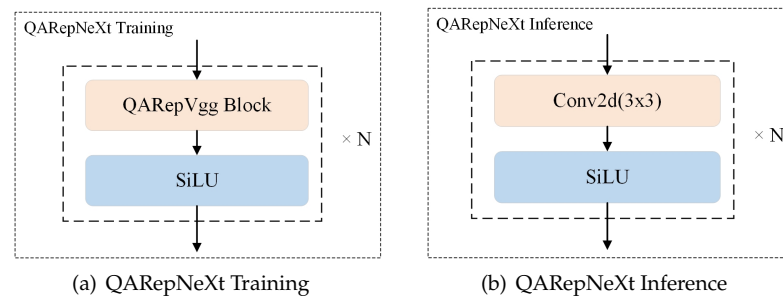


Figure 6. Illustration of the QARepNeXt module, which adopts a QARepVgg block (multi-branch structure) during training and a convolution module (single-branch structure) during inference.

2.2.2. Attention Mechanism in the Neck Network

Compared with the C3 structure, QARepNeXt has relatively fewer parallel branches, which slightly diminishes the model's representational capacity. Therefore, we applied the triplet attention mechanism [29] to the fused feature maps in the neck network [30] to enhance the model's feature extraction capabilities.

The triplet attention modules calculated the attention weights by capturing the interactions between different dimensions of the input tensor. It fuses three parallel branches for the (channel, height), (channel, width), and (height, width) dimensions, as shown in Figure 7. The Z-pool layer combines the average pooled and max pooled features across each dimension to reduce the tensor's zeroth dimension to two. This can obtain a detailed tensor representation while reducing its depth and making subsequent computations more efficient. The mathematical representation of *Z-Pool* can be defined as

$$Z\text{-Pool}(x) = [\text{MaxPool}_{0d}(x), \text{AvgPool}_{0d}(x)]. \quad (1)$$

We only applied the triple attention mechanism to the last two layers of the neck network, which contain higher-level semantic features, as depicted in Figure 3. This could improve the detection ability without significantly increasing the detection latency.

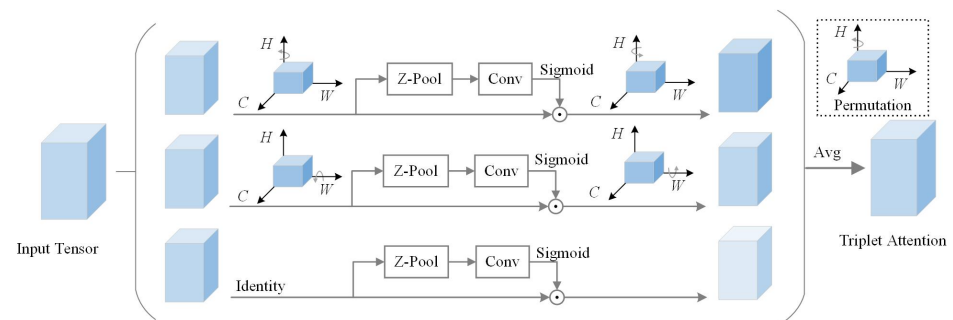


Figure 7. Illustration of the triplet attention, which has three parallel branches.

2.2.3. Loss Function

In our field strawberry detection scenario, some instances were difficult to detect due to lighting variations, leaf occlusions, and similar colors between the foreground and

background. To address this, Lin et al. [31] proposed the focal loss function, which can mitigate the foreground–background class imbalance problem and enhance the learning of challenging cases. This approach significantly improved the detection accuracy.

The focal loss function reduces the weight assigned to well-classified instances of loss by reshaping the standard cross-entropy loss, which is defined as

$$CE(p, y) = \begin{cases} -\log(p) & \text{if } y = 1 \\ -\log(1 - p) & \text{otherwise,} \end{cases} \quad (2)$$

where y is a component of $\{\pm 1\}$ that indicates the ground truth, and $p \in [0, 1]$ demonstrates the probability for the class labeled as $y = 1$. To simplify our notation, let us define p_t as follows:

$$p_t = \begin{cases} p & \text{if } y = 1 \\ 1 - p & \text{otherwise,} \end{cases} \quad (3)$$

and the CE loss can be rewritten as

$$CE(p, y) = CE(p_t) = -\log(p_t). \quad (4)$$

The class imbalance problem is typically addressed by introducing a weight parameter α within the range $[0, 1]$ for class 1 and $1 - \alpha$ for class -1 . With the definition of α_t being the same as p_t , the α -balanced CE loss can be written as

$$CE(p_t) = -\alpha_t \log(p_t). \quad (5)$$

Although α can balance the importance of positive and negative examples, it cannot differentiate between easy and hard examples. To address this, the focal loss function introduces a modulating factor $(1 - p_t)^\gamma$ to reduce the importance of easy examples and direct the training toward hard examples. The focal loss function is expressed as follows:

$$FL(p_t) = -\alpha_t (1 - p_t)^\gamma \log(p_t), \quad (6)$$

where the tunable focusing parameter $\gamma > 0$.

The modulating factor effectively reduces the impact of loss caused by easy examples. For instance, if $\gamma = 2$, an example with a classification of $p_t = 0.95$ will demonstrate 400× lower loss compared with CE. Consequently, there was an increased significance in rectifying misclassified examples.

3. Experiments And Results

3.1. Experimental Setting

3.1.1. Parameters And Equipment

We conducted the experiments with a batch size of 16, an image size of 640×640 pixels, and 300 epochs. The remaining parameters were kept as the default values of YOLOv5s. The runtime environment is shown in Table 2.

Table 2. The experimental environment used in the current study.

Hardware and Software	Configuration
CPU	Intel(R) Core(TM) i9-11900K
RAM	64 GB
GPU	GeForce RTX 2080Ti
Operating system	Windows 10
Cuda	Cuda 11.3.0
Data processing	Python 3.10
Deep learning framework	Pytorch 1.9.0

3.1.2. Index Definition

Various metrics were employed to measure the effectiveness of RTF-YOLO, namely, recall, precision, AP (average precision), mAP (mean average precision), parameter count, FPS, and FLOPs (floating point operations). The formulas for calculating these performance metrics are as follows:

$$R = \frac{TP}{TP + FN}, \quad (7)$$

$$P = \frac{TP}{TP + FP}, \quad (8)$$

$$F1 = \frac{2 \times P \times R}{P + R}, \quad (9)$$

$$AP = \int_0^1 P(R) dR, \quad (10)$$

$$mAP = \frac{\sum_{i=1}^n AP_i}{n}, \quad (11)$$

where TP represents the count of accurate predictions the model makes for positive examples. In contrast, FP denotes the count of negative examples incorrectly identified as positive. FN signifies the measure of positive samples erroneously categorized as negative. $F1$ is a composite evaluation index of P and R . AP is the area under the P - R (precision–recall) curve, indicating the overall model performance. A larger area implies better performance. mAP , the mean value of AP across all classes, indicates the model's detection performance. FLOPs refer to the computational complexity measurement of a model, while the detection speed is evaluated with FPS, which represents the number of images the model can process per second during inference.

3.2. Experimental Results and Analysis

This paper proposes three improved network models derived from YOLOv5: R-YOLO, RT-YOLO, and RTF-YOLO. R-YOLO only utilized the QARepNeXt module for improvement. RT-YOLO incorporated both the QARepNeXt module and the triplet attention mechanism for its enhancement. Meanwhile, RTF-YOLO combined the QARepNeXt module, the triplet attention mechanism, and the focal loss function for improvement. Several experiments were conducted to validate the improved models' effectiveness.

3.2.1. Experiments of the QARepNeXt Module

To validate the effectiveness of the QARepNeXt module, several improved C3 modules based on mainstream lightweight networks, such as GhostNetV2, MobileNetV3, and FasterNet, were constructed to replace the original C3 module in YOLOv5. Table 3 presents the experimental results of the different models.

Table 3. Performance, complexity, and analysis speed of various backbones.

Backbone	$mAP_{0.5}$ (%)	$mAP_{0.75}$ (%)	$mAP_{0.5:0.95}$ (%)	Params (M)	FLOPs (G)	FPS
YOLOv5	86.55	62.08	55.61	7.03	16.0	122
GhostNetV2	87.05	58.53	53.95	4.90	10.6	114
MobileNetV3	86.77	58.74	54.63	6.44	14.1	122
ShuffleNetV2	85.52	61.39	53.72	4.85	10.3	97
InceptionNet	87.10	63.72	55.61	5.37	11.7	94
FasterNet	87.34	64.82	57.44	5.72	12.6	120
QARepNeXt	86.58	65.46	56.87	13.00	33.6	154

We strived to find a fast backbone network to meet the need for real-time strawberry detection. Among various metrics, the actual detection speed, that is, FPS, served as our

primary selection criterion for the backbone. According to the results in Figure 3, these lightweight models effectively reduced the model's parameter count and FLOPs. However, these improved models, except for QARepNeXt, did not demonstrate performance improvements that matched the reduction in FLOPs. The detection speed was determined by both FLOPs and FLOPS (floating point operations per second) [28], which is captured by

$$\text{FPS} = \frac{\text{FLOPs}}{\text{FLOPS}} \quad (12)$$

It is important to achieve higher FLOPS beyond simply reducing FLOPs for faster neural networks. While many attempts have been made to reduce FLOPs, such as reducing the parameters, they seldom consider optimizing FLOPS simultaneously to achieve truly low latency. The QARepNeXt converts to a single-path structure during inference, while other models all have complex multi-branch designs. The single-path structure is a hardware-efficient architecture that can utilize computing ability and memory bandwidth more effectively [26], which results in higher FLOPS. Despite having higher parameters and FLOPs, the QARepNeXt achieved the best detection speed. In conclusion, we selected the QARepNeXt as the primary module for the backbone and neck, which efficiently improved the detection speed without compromising accuracy.

3.2.2. Experiments of the Triplet Attention Mechanism

The attention mechanisms were embedded in the last two layers of R-YOLO's neck network to better extract crucial features. We conducted comparative experiments on various types of attention mechanisms, including channel attention mechanisms, such as SE, and channel and spatial attention mechanisms, such as CBAM and the triplet. The experimental results for different attention mechanisms are shown in Table 4.

According to the results in Table 4, all the attention mechanisms improved the detection accuracy of R-YOLO. Given an input image, the channel attention focused on what was meaningful. It considered each channel of a feature map as a feature detector, contributing more to the classification. In this study, the primary challenge was to accurately identify the target class, and thus, the channel attention contributed more to the model's accuracy. This explained why the SE attention mechanism, which only included the channel attention part, achieved a significant performance improvement [32]. Contrarily, the spatial attention component complemented the channel attention by emphasizing the precise localization of informative parts, which led to further detection accuracy improvements. As a result, the channel and spatial attention mechanisms achieved a better performance than SE. Since the triplet attention can capture the cross-dimensional interaction of the channel and spatial dimension, it achieved the highest detection precision rate, $mAP_{0.5}$, $mAP_{0.75}$, and $mAP_{0.5:0.95}$ metrics. Therefore, we selected the triplet attention mechanism to enhance the model's feature extraction.

Table 4. Performance comparison of R-YOLO with various attention mechanisms.

Model	P (%)	R (%)	$mAP_{0.5}$ (%)	$mAP_{0.75}$ (%)	$mAP_{0.5:0.95}$ (%)	FPS
R-YOLO	89.05	78.50	86.58	62.75	54.87	154
R-YOLO + SE	89.69	79.44	88.02	62.24	55.35	149
R-YOLO + CBAM	89.49	81.26	88.54	62.07	55.80	137
R-YOLO + coordinate	89.96	80.48	88.24	62.14	55.48	141
R-YOLO + NAM	89.84	80.40	88.10	62.36	55.06	152
R-YOLO + SGE	89.37	80.02	87.82	62.78	54.93	141
R-YOLO + SimAM	90.34	80.52	89.02	63.19	56.07	139
R-YOLO + triplet	91.43	81.05	89.21	63.46	56.29	145

3.2.3. Experiments of the Focal Loss Function

The focal loss function was integrated with RT-YOLO to improve the recall rate by enhancing the model's learning ability toward challenging examples. The focal loss function has two key parameters, α and γ . In our experiment, we explored various values of α and γ , where $\alpha \in [0.1, 0.2]$ and $\gamma \in [1, 5]$, and found that the optimal result was achieved when $\alpha = 0.13$ and $\gamma = 1.4$ on the validation dataset, as shown in Table 5. We applied the best parameter values on the test dataset and achieved an improved $mAP_{0.5}$ of 90.24%, which manifested the model's robustness. A lower value of α (0.13 in this study) is typically chosen in conjunction with a higher value of γ (1.4 in this study) [31]. The results of our research are in agreement with this assertion.

Table 5. Performance ($mAP_{0.5}(\%)$) of the focal loss function under varying α and γ parameters.

$\alpha \backslash \gamma$	1.00	1.20	1.40	1.60	2.00	2.40	3.00	5.00
0.10	88.26	87.72	89.63	88.28	88.82	82.96	73.98	51.57
0.11	87.41	88.31	89.97	87.77	88.48	83.18	75.11	52.33
0.12	88.37	88.36	90.16	89.08	87.66	84.17	72.32	55.10
0.13	88.81	89.60	90.32	88.04	88.57	84.88	76.97	54.84
0.15	89.67	88.67	89.84	89.06	88.61	83.59	79.63	53.91
0.20	88.75	88.27	88.31	87.66	86.79	85.95	83.90	53.07

3.3. Ablation Experiments of Different Improved Models

The detection results of different improved models are shown in Table 6 and Figure 8. R-YOLO improved the detection speed by 32 FPS while maintaining the $mAP_{0.5}$ value essentially unchanged. Compared with R-YOLO, RT-YOLO improved the $mAP_{0.5}$ by 2.6 points, but the attention mechanism introduced an increase in FLOPs, which resulted in a slight reduction in the detection speed. Finally, RTF-YOLO integrated the focal loss function with RT-YOLO, which improved the model's recall rate and improved by more than 1 $mAP_{0.5}$ point. Ultimately, compared with the original YOLOv5 model, our proposed RTF-YOLO model achieved an almost 3.6 point $mAP_{0.5}$ improvement and a 23 FPS increase in detection speed.

The confusion matrix of different improved models is shown in Figure 9. In the confusion matrix, the horizontal axis represents the ground truth labels, while the vertical axis represents the labels predicted by the model. The first two columns represent the counts of the ripen and unripen strawberry labels, respectively. Their sum equalled the total number of labels in the test dataset in Table 1, which was 820. The triplet attention mechanism boosted the feature extraction, which raised the count of accurately detected strawberries by 32. Unripened strawberries were susceptible to false detections and missed detections for resembling leaves in color. The focal loss function could enhance the learning capacity for these challenging objects. As a result, RTF-YOLO reduced the count of unripened strawberries' false and missed detections by six compared with RF-YOLO.

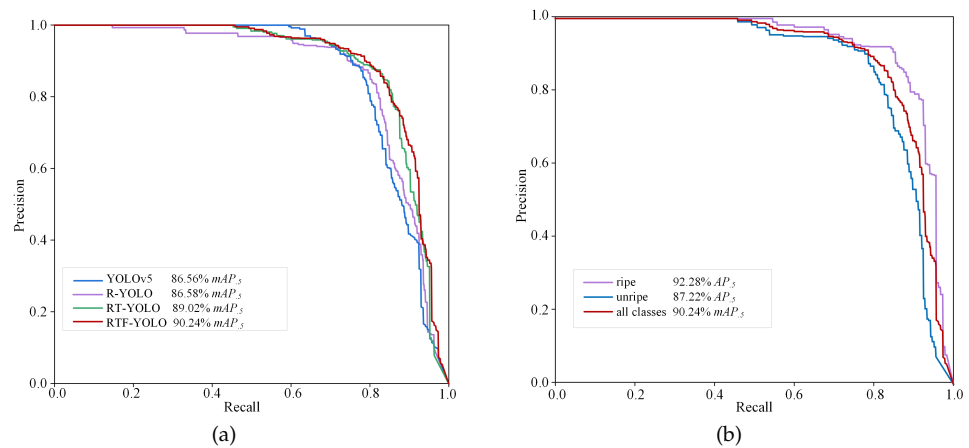


Figure 8. The precision–recall curves. (a) The recall–precision curve of different improved models. (b) The recall–precision curve of various categories obtained by RTF-YOLO.

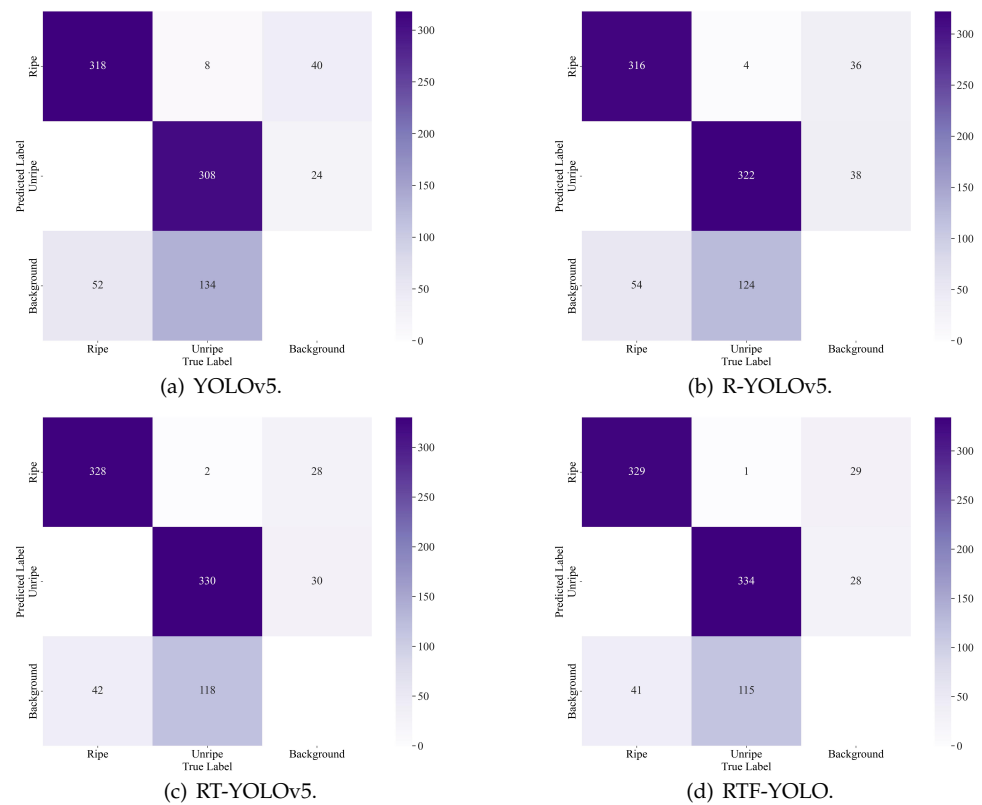


Figure 9. Comparative confusion matrices of various improved models.

Table 6. The ablation experiment results of different improved models.

Model	P (%)	R (%)	$AP_{0.5}$ (%)		$mAP_{0.5}$ (%)	$mAP_{0.75}$ (%)	$mAP_{0.5:0.95}$ (%)	FPS
			Unripen	Ripen				
YOLOv5	89.67	77.21	82.60	90.50	86.55	62.08	55.61	122
R-YOLO	89.05	78.50	82.20	90.90	86.58	62.75	54.87	154
RT-YOLO	91.43	81.05	84.60	93.50	89.21	63.46	56.29	145
RTF-YOLO	91.92	81.43	86.30	94.30	90.24	64.84	57.65	145

The visualization results of different improved models are depicted in Figure 10. The $r1$ label in the diagram signifies unripened strawberries, whereas the $r2$ label signified

ripened strawberries. All the improved models demonstrated some enhancements in the occurrences of false positives and negatives. However, considering the overall performance, RTF-YOLO achieved the best results. For example, as shown in (a), the YOLOv5 model incorrectly detected the leaves in the bottom-right and right sides as unripened strawberry fruits. The RT-YOLO model resolved the false positive problem but introduced a false negative in the top right. RTF-YOLO, on the other hand, successfully addressed both types of problems. As shown in (b), YOLOv5 and R-YOLO suffered from false negatives for unripened strawberries, which had colors similar to the leaves. However, RTF-YOLO effectively eliminated false negatives by enhancing the learning of such challenging examples.

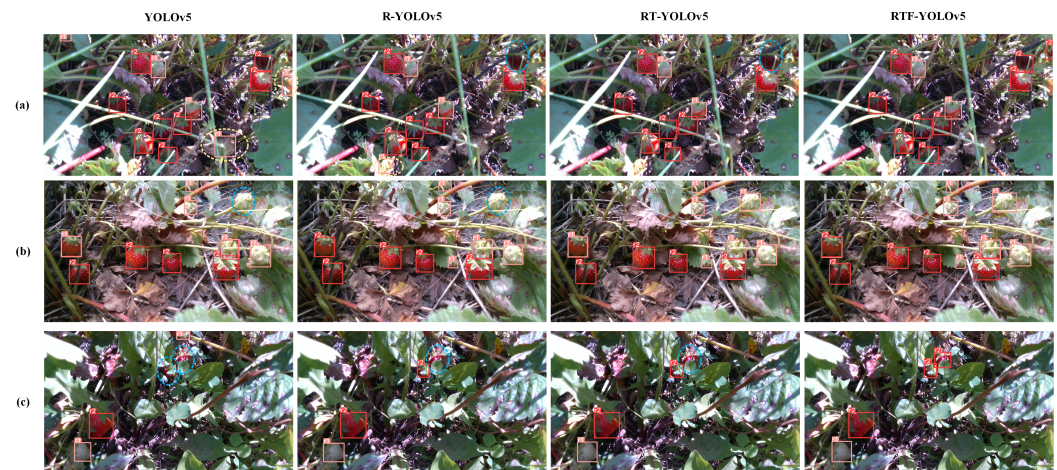


Figure 10. Visualization results of YOLOv5, R-YOLO, RT-YOLO, and RTF-YOLO are shown in their respective columns, with subfigures (a–c) serving as three examples. The false positive samples are encircled with a yellow dashed line, while the false negative samples are encircled with a blue dashed line.

Comparative Experimental Analysis of Different Models

We compared RTF-YOLO with other mainstream target detection networks, including YOLOv5s, YOLOv5n, ASFF-YOLOv5 [22], DSE-YOLOv5 [19], DSW-YOLOv5 [8], and Faster R-CNN, as shown in Table 7. According to the experimental results, Faster R-CNN achieved a recall rate of 81.04%, but its precision rate was only 38.43%, which led to many false detections of strawberries. In contrast, SSD achieved a precision rate of 82.33%, but its recall rate was only 19.24%, which led to many missed detections. Faster RCNN employed a single high-level feature map for target classification and localization, which led to its inadequate detection ability for small- and multi-scale targets. Although SSD used multi-scale feature maps from different layers, it failed to fully exploit the information from low-level high-resolution feature maps, which is vital for small object detection. As a result, neither of the two networks could meet the requirements for multi-stage strawberry detection.

Table 7. Comparative analysis of performance and efficiency metrics for various object detection models.

Model	P (%)	R (%)	mAP _{0.5} (%)	Params (M)	FPS
Faster R-CNN	38.43	81.04	64.32	137.10	23
SSD	82.33	19.24	63.24	26.29	50
YOLOv5	89.67	77.21	86.65	7.03	122
ASFF-YOLOv5	86.00	81.75	87.25	12.46	100
DSE-YOLO	85.34	81.09	87.98	224.39	21
DSW-YOLO	82.80	82.10	86.70	32.40	42
RTF-YOLO	91.92	81.43	90.24	13.00	145

YOLOv5 utilized the PANet network to integrate feature maps from diverse levels, which enhanced the receptive field and achieved a richer input representation. Models based on YOLOv5 achieved better detection accuracy. The $mAP_{0.5}$ metric for RTF-YOLO was 90.24%, which showed improvements of 3.6% and 3% compared with YOLOv5s and ASFF-YOLOv5, respectively. YOLOv5's single-stage detection scheme offered a higher detection speed. However, introducing enhancement modules, such as the ASFF module, DSE module, and attention mechanism, increased the model complexity and reduced the detection speed. Benefiting from RTF-YOLO's improved backbone network, its detection speed surpassed YOLOv5s and other YOLO-based improved networks by over 19% and 45%, respectively. Although DSE-YOLO and DSW-YOLO achieved a high detection accuracy, the large model sizes and slower detection speeds limit their deployment on edge computing devices. In contrast, RTF-YOLO offered a threefold advantage in detection speed and model scale. Therefore, RTF-YOLO was more suitable for detecting strawberries in complex environments in real time.

4. Conclusions

This article proposes a novel RTF-YOLO network model for strawberry detection under fluctuating lighting and fruit occlusion scenarios. The RTF-YOLO model was obtained by integrating YOLOv5 with the QARepNeXt module, the triplet attention mechanism, and the focal loss function. The QARepNeXt module improved the detection speed of the model, while the triplet attention mechanism enhanced the extraction capability. The focal loss function was utilized to address the foreground–background class imbalance problem and enhance the learning ability of challenging examples. The model's performance was validated with a dataset collected from the field. The experimental results showed that the model achieved a precision of 91.92%, a recall rate of 81.43%, an mAP of 90.24%, and a detection speed of 145 FPS. Compared with other mainstream object detection algorithms, RTF-YOLO was more advantageous in terms of the mAP , model size, and detection speed. Consequently, the proposed algorithm can provide guidance for the yield prediction and automated harvesting of strawberries.

Author Contributions: Conceptualization, methodology, and software, S.S.; data curation, F.D. and Z.T.; writing—review and editing, S.S., F.D., Z.T. and C.H.; supervision and project administration, F.D., Z.T. and C.H. All authors read and agreed to the published version of this manuscript.

Funding: This research was funded by the Chengdu Agricultural Science and Technology Center Local Finance Special Fund Project (NASC2021ST02) and Sichuan Provincial Science and Technology Plan Project (2023YFSY0052).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available upon request from the corresponding authors.

Conflicts of Interest: Author Chunxiao Han was employed by the company SDIC Xinjiang Luobupo Potash Co., Ltd. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Manganaris, G.A.; Goulas, V.; Vicente, A.R.; Terry, L.A. Berry antioxidants: Small fruits providing large benefits. *J. Sci. Food Agric.* **2014**, *94*, 825–833. [\[CrossRef\]](#)
2. Zhou, C.; Hu, J.; Xu, Z.; Yue, J.; Ye, H.; Yang, G. A novel greenhouse-based system for the detection and plumpness assessment of strawberry using an improved deep learning technique. *Front. Plant Sci.* **2020**, *11*, 559. [\[CrossRef\]](#)
3. Zhao, Y.; Gong, L.; Huang, Y.; Liu, C. A review of key techniques of vision-based control for harvesting robot. *Comput. Electron. Agric.* **2016**, *127*, 311–323. [\[CrossRef\]](#)
4. Linker, R.; Cohen, O.; Naor, A. Determination of the number of green apples in RGB images recorded in orchards. *Comput. Electron. Agric.* **2012**, *81*, 45–57. [\[CrossRef\]](#)

5. Arefi, A.; Motlagh, A.M. Development of an expert system based on wavelet transform and artificial neural networks for the ripe tomato harvesting robot. *Aust. J. Crop Sci.* **2013**, *7*, 699–705.
6. Lu, J.; Sang, N. Detecting citrus fruits and occlusion recovery under natural illumination conditions. *Comput. Electron. Agric.* **2015**, *110*, 121–130. [[CrossRef](#)]
7. Hamuda, E.; Mc Ginley, B.; Glavin, M.; Jones, E. Improved image processing-based crop detection using Kalman filtering and the Hungarian algorithm. *Comput. Electron. Agric.* **2018**, *148*, 37–44. [[CrossRef](#)]
8. Du, X.; Cheng, H.; Ma, Z.; Lu, W.; Wang, M.; Meng, Z.; Jiang, C.; Hong, F. DSW-YOLO: A detection method for ground-planted strawberry fruits under different occlusion levels. *Comput. Electron. Agric.* **2023**, *214*, 108304. [[CrossRef](#)]
9. Kamilaris, A.; Prenafeta-Boldú, F.X. Deep learning in agriculture: A survey. *Comput. Electron. Agric.* **2018**, *147*, 70–90. [[CrossRef](#)]
10. Zhang, Y.; Yu, J.; Chen, Y.; Yang, W.; Zhang, W.; He, Y. Real-time strawberry detection using deep neural networks on embedded system (rtsd-net): An edge AI application. *Comput. Electron. Agric.* **2022**, *192*, 106586. [[CrossRef](#)]
11. Yu, Y.; Zhang, K.; Liu, H.; Yang, L.; Zhang, D. Real-time visual localization of the picking points for a ridge-planting strawberry harvesting robot. *IEEE Access* **2020**, *8*, 116556–116568. [[CrossRef](#)]
12. Mejia, G.; de Oca, A.M.; Flores, G. Strawberry localization in a ridge planting with an autonomous rover. *Eng. Appl. Artif. Intell.* **2023**, *119*, 105810. [[CrossRef](#)]
13. Yu, Y.; Zhang, K.; Yang, L.; Zhang, D. Fruit detection for strawberry harvesting robot in non-structural environment based on Mask-RCNN. *Comput. Electron. Agric.* **2019**, *163*, 104846. [[CrossRef](#)]
14. Tang, C.; Chen, D.; Wang, X.; Ni, X.; Liu, Y.; Liu, Y.; Mao, X.; Wang, S. A fine recognition method of strawberry ripeness combining Mask R-CNN and region segmentation. *Front. Plant Sci.* **2023**, *14*, 1211830. [[CrossRef](#)]
15. Perez-Borrero, I.; Marin-Santos, D.; Gegundez-Arias, M.E.; Cortes-Ancos, E. A fast and accurate deep learning method for strawberry instance segmentation. *Comput. Electron. Agric.* **2020**, *178*, 105736. [[CrossRef](#)]
16. Chen, Y.; Lee, W.S.; Gan, H.; Peres, N.; Fraisse, C.; Zhang, Y.; He, Y. Strawberry yield prediction based on a deep neural network using high-resolution aerial orthoimages. *Remote Sens.* **2019**, *11*, 1584. [[CrossRef](#)]
17. Zheng, C.; Liu, T.; Abd-Elrahman, A.; Whitaker, V.M.; Wilkinson, B. Object-Detection from Multi-View remote sensing Images: A case study of fruit and flower detection and counting on a central Florida strawberry farm. *Int. J. Appl. Earth Obs. Geoinf.* **2023**, *123*, 103457. [[CrossRef](#)]
18. Li, J.; Zhu, Z.; Liu, H.; Su, Y.; Deng, L. Strawberry R-CNN: Recognition and counting model of strawberry based on improved faster R-CNN. *Ecol. Inform.* **2023**, *77*, 102210. [[CrossRef](#)]
19. Wang, Y.; Yan, G.; Meng, Q.; Yao, T.; Han, J.; Zhang, B. DSE-YOLO: Detail semantics enhancement YOLO for multi-stage strawberry detection. *Comput. Electron. Agric.* **2022**, *198*, 107057. [[CrossRef](#)]
20. Zhou, X.; Lee, W.S.; Ampatzidis, Y.; Chen, Y.; Peres, N.; Fraisse, C. Strawberry maturity classification from UAV and near-ground imaging using deep learning. *Smart Agric. Technol.* **2021**, *1*, 100001. [[CrossRef](#)]
21. Chai, J.J.; Xu, J.L.; O'Sullivan, C. Real-Time Detection of Strawberry Ripeness Using Augmented Reality and Deep Learning. *Sensors* **2023**, *23*, 7639. [[CrossRef](#)] [[PubMed](#)]
22. Li, Y.; Xue, J.; Zhang, M.; Yin, J.; Liu, Y.; Qiao, X.; Zheng, D.; Li, Z. YOLOv5-ASFF: A Multistage Strawberry Detection Algorithm Based on Improved YOLOv5. *Agronomy* **2023**, *13*, 1901. [[CrossRef](#)]
23. Wang, J.; Perez, L. The effectiveness of data augmentation in image classification using deep learning. *Convolutional Neural Netw. Vis. Recognit.* **2017**, *11*, 1–8.
24. Lin, T.Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature pyramid networks for object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 2117–2125.
25. Liu, S.; Qi, L.; Qin, H.; Shi, J.; Jia, J. Path aggregation network for instance segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 8759–8768.
26. Ding, X.; Zhang, X.; Ma, N.; Han, J.; Ding, G.; Sun, J. Repvgg: Making vgg-style convnets great again. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 13733–13742.
27. Chu, X.; Li, L.; Zhang, B. Make RepVGG Greater Again: A Quantization-aware Approach. *arXiv* **2022**, arXiv:2212.01593.
28. Chen, J.; Kao, S.H.; He, H.; Zhuo, W.; Wen, S.; Lee, C.H.; Chan, S.H.G. Run, Don't walk: Chasing higher FLOPS for faster neural networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 17–24 June 2023; pp. 12021–12031.
29. Misra, D.; Nalamada, T.; Arasanipalai, A.U.; Hou, Q. Rotate to attend: Convolutional triplet attention module. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, Virtual, 5–9 January 2021; pp. 3139–3148.
30. Li, H.; Li, J.; Wei, H.; Liu, Z.; Zhan, Z.; Ren, Q. Slim-neck by GSConv: A better design paradigm of detector architectures for autonomous vehicles. *arXiv* **2022**, arXiv:2206.02424.
31. Lin, T.Y.; Goyal, P.; Girshick, R.; He, K.; Dollár, P. Focal loss for dense object detection. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2980–2988.
32. Woo, S.; Park, J.; Lee, J.Y.; Kweon, I.S. Cbam: Convolutional block attention module. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 3–19.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.